# به نام خدا

## توضیحات فاز سوم پروژه برنامه نویسی پیشرفته

محمد علی علما

# قسمت اول : منابع مورد استفاده

**برای ذخیره سازی فایل در ساختار جیسون ، از کتابخانه های** Jackson، Gson و json.simple **به طور ترکیبی استفاده شده است .**

**از اطلاعات سایت های زیر بهره برده شده است :**

۱- https://stackoverflow.com/
۲- https://www.geeksforgeeks.org/
۳- https://www.tutorialspoint.com/
۴- http://javapro.ir/
۵- http://www.java2s.com/
۶- http://hearthcards.net/
۷- https://hearthstone.gamepedia.com/
۸- **طراحی سیستم** Request – Response **:** https://stackoverflow.com/questions/18140964/return-different-type-of-data-from-a-method-in-java/49216997

**همچنین از راهنمایی های افراد زیر نیز بهره برده شده است:**

۱- **استاد مجتبی استواری**
۲- **استاد حسین بومری**
۳- **فرزین نصیری**
۴- **امیرمحمد شعبانی**
۵- **سید امیر محمد سادات شکوهی**

# قسمت دوم : ساختار کد به همراه نقاط قوت و ضعف

**ساختار کلی برنامه شبیه فاز دو است اما تغییرات گسترده ای در بازی ایجاد شده است :**

۱- **برای هر کدام از کارت های بازی ، یک کلاس جداگانه درست شد . اما همچنان اطلاعات هر کارت از فایل های جیسون خارج از برنامه نگه داری و لود میشود . دلیل ایجاد کلاس برای هر کارت ، استفاده بهینه از دیزاین پترن ویزیتور بوده است . ( در ادامه توضیح داده میشود )**
۲- **دیزاین پترن ویزیتور : برای اکشن کارت ها ، و همچنین قدرت هیرو های بازی ، از دیزاین پترن ویزیتور استفاده کردم . این دیزاین پترن به طور فوق العاده ای ، لاجیک را از مدل خارج کرد و همچنین باعث شد ،در صورت اضافه شدن ویژگی جدیدی به یک کارت و یا افزودن کارت با ویژگی های خاص تر ، بتوان به راحتی این کار را انجام داد بدون اینکه به کلاس های کارت ( کلاس های مدل ) تغییری اضافه کرد.**

۳- **سیستم** Request – Response **: برا جدا کردن گرافیک از لاجیک بازی ، سیستمی طراحی شد که اطلاعات مختلفی را با دیتاتایپ های مختلف از لاجیک برنامه دریافت میکند ( به درخواست گرافیک ) و آن را در اختیار گرافیک قرار میدهد . این سیستم ، به خوبی گرافیک و لاجیک را از یکدیگر جدا کرده است ( البته در بعضی قسمت ها همچنان وابستگی هایی وجود دارد ، برای مثال در بعضی قسمت ها ، از گرافیک برنامه به طور مستقیم از توابع کلاس** admin **که واسط بین گرافیک و لاجیک برنامه است ، استفاده میکنم . یا در چند تابع از کلاس** admin **، توابعی از گرافیک بازی ( بیشتر مربوط به** BoardPanel **) را صدا میزنم . )**

۴- **پکیج بندی بازی بهینه تر شده است .**

۵- **کلاس** Admin **وظیفه ارتباط بین لاجیک و گرافیک بازی را برعهده دارد .**

۶- **کلاس** game manager **که حالت های مختلف بازی (** normal mode , deck reader , practice **) را میسازد ( با استفاده از سازنده های مختلف ) و استیت کلی بازی در حال اجرا را نگهداری میکند.**

۷- **از نقاط ضعف بازی ، توسعه تایمر ها و ماوس لیستنر ها به صورت اینرکلس در کلاس** BoardPanel **است که باعث افزایش حجم کد این کلاس شده است .**

۸- **نقطه ضعف دیگر برنامه ، همان سیستم** Request – Response **است که به صورت سوییچ کیس طراحی شده است .**

۹- **مورد لازم به ذکر دیگر ، چرخش ۱۸۰ درجه ای صفحه بازی در حالت های** Normal mode **و** DECK READER **است که دلیل آن را در قسمت بعد شرح میدهم .**

## قسمت سوم :ارائه دلیل برای ساختار کد

**شاید مهم ترین قسمت کد ، طراحی اکشن های کارت ها به وسلیه دیزاین پترن ویزیتور باشد . دلیل استفاده از این دیزاین پترن ، خارج کردن لاجیک از مدل ها و همچنین امکان اضافه کردن قابلیت های جدید به کارت ها ، بدون ایجاد تغییر در ساختار کلاس های مدل ها است .**

**همچنین در حالت های** normal mode **و** Deck reader **، در ابتدای هر نوبت ، صفحه بازی ۱۸۰ درجه میچرخد ( این چرخش به صورت چرخیدن واقعی گرافیک نیست ، بلکه در انتهای هر نوبت ، در لاجیک بازی ، جای بازیکن دوست و دشمن با یکدیگر عوض میشود و همین امر موجب میشود که به صورت خودکار هر بازیکنی که نوبت او است ، در پایین صفحه قرار بگیرد . ) دلیل اینکار ، متمرکز کردن فعالیت ماوس لیستنر ها در نیمه پایین صفحه است . بدین ترتیب ، اکثر قابلیت ها و کارهای ماوس لیستنر ها در نیمه پایین صفحه خواهد بود و تنها قسمتی که ماوس لیستنر در نیمه بالایی نیز کار میکند ، بزرگ کردن کارت های پلی شده بازیکن مقابل با هاور کردن ماوس روی آن کارت است .**

**برای جدا کردن لاجیک از گرافیک نیز ، سیستیم** Request-Response **طراحی شده است . این سیستم کمک میکند که وابستگی مستقیم گرافیک به لاجیک و بر عکس به حداقل برسد و تنها از طریق** admin **این کار میسر خواهد بود . با ایجاد این سیستم ، در فاز بعد با اضافه شدن شبکه ، کار برای توسعه بازی راحت تر خواهد بود و با انتقال لاجیک بازی به سرور ، میتوان با این سیستم درخواستی را به سرور ارسال و جواب آن را دریافت کرد .**

## قسمت چهارم : کتابخانه ها و ورژن ها

**جاوا : نسخه ۸**

Gson **: نسخه ۲.۲.۲**

Jackson **: نسخه ۱.۹.۹**

Json.simple **: نسخه ۱.۱.۱**

Jlayer **: نسخه ۱.۰.۱ ( فایل های صوتی )**

**برای راحتی استفاده از این کتابخانه ها ، پروژه در قالب** gradle **ایجاد شده است .**